A Multitude of Linguistically-rich Features for Authorship Attribution

Ludovic Tanguy, Assaf Urieli, Basilio Calderone, Nabil Hathout, Franck Sajous

A multitude of linguistically-rich features for authorship attribution

Ludovic Tanguy, Assaf Urieli, Basilio Calderone, Nabil Hathout, and Franck Sajous

CLLE-ERSS: CNRS & University of Toulouse *firstname.lastname*@univ-tlse2.fr

Abstract. This paper reports on the procedure and learning models we adopted for the 'PAN 2011 Author Identification' challenge targetting real-world email messages. The novelty of our approach lies in a design which combines shallow characteristics of the emails (words and trigrams frequencies) with a large number of *ad hoc* linguistically-rich features addressing different language levels. For the author attribution tasks, all these features were used to train a maximum entropy model which gave very good results. For the single author verification tasks, a set of features exclusively based on the linguistic description of the emails' messages was considered as input for symbolic learning techniques (rules and decision trees), and gave weak results. This paper presents in detail the features extracted from the corpus, the learning models and the results obtained.

1 Introduction and motivations

Given a set of texts and the corresponding authors (in our case email messages), the basic idea underlying Authorship Attribution (AA) systems is that various calculable textual features may be relevant enough to capture statistically-based generalizations of the 'stylistic distinctiveness' of the writers, in order to distinguish texts written by different authors. These 'textual features' thus refer to the ability of representing a text by a set of attributes that capture a particular author's 'writing style' [20].

As members of the NLP team of a linguistics research laboratory, we decided to concentrate on calculating a large set of features, as we saw in this challenge an opportunity to take advantage of our experience in a wide variety of automated text annotation tasks. As will be presented in detail in § 2, we addressed both traditional features used for AA [10, 18], and more innovative ones corresponding to the investigation of more focused textual phenomena.

Our definition and selection of these features was based on an approach that originates in the linguistic side of NLP and computational linguistics. More precisely, the first phase of our work was to carefully examine the data, and to use a combination of human intuition and computer-aided investigation tools, a common method used in areas such as corpus linguistics.

The next step was to design (in most cases, to recycle) specific processing methods to compute the selected features. Team work, available linguistic resources and the variety of our experiences in different areas of computational linguistics were crucial assets here. As described in the following sections, we were thus able to combine morphological, syntactic and semantic techniques, and also to design many data-specific features. If some of the features rely on well-known techniques, we also designed innovative approaches, such as the different measures used to assess syntactic complexity and semantic cohesion.

The last step was to feed all these features into a machine learning algorithm (§ 3). Given the sheer number of features (especially when word and trigrams frequencies are involved), our choice was to use a maximum entropy learner, which is able to cope with such quantities and with a variety of feature types (both quantitative and qualitative). However, we also wanted to have some kind of feedback of this procedure, and to identify which features were the most efficient for the task. This led us to apply other machine learning methods, namely tree- and rule-based algorithms whose main advantage is to give intelligible representations of the learned schemes. As far as the task results are concerned, this second choice was not a wise move, as we will discuss in the last section.

The PAN 2011 Authorship Attribution competition was based on real-world email messages (extracted from the Enron corpus) and consisted in two different sets of tasks. The first one (standard authorship *attribution*) required the participants to infer who was the author of each message from the test data (from all the authors present in the training data). Some of the test sets also contained emails written by unknown authors (absent from the training data): these subtasks' names are identified with a + sign. Training sets contained 3,000 messages from 26 authors (*small* data set) and 10,000 messages from 72 authors (*large* data set), while test sets ranged from 400 to 1,500 messages. The second set of tasks (authorship *verification*) focused on a single target author, requiring the participant to simply decide for each message in the test data if it was written by this particular author or not. Three different authors were targeted: for each author the training data contained about 50 messages, and test data about 100, with of course a totally unknown distribution.

The data itself was very challenging, especially when compared to other data sets used for this kind of task (mostly litterary texts). In addition to the (very) sloppy writing inherent to email messages, the data was very heterogeneous, as the authors used emails in very different contexts (personal as well as formal and informal professional communication). Some emails even contained non-english text, raw computer data and automatically generated content.

2 Linguistic features

In this section we give a detailed list of all the features used in this experiment. We chose to group them according to the linguistic units they address. We identified four such levels: sub-word (morphological units, character trigrams), word, sentence or phrase (syntax) and message.

2.1 Preprocessing

As many of the features used in this experiment required linguistic information from different language levels, our first step was to apply NLP tools to the raw text messages provided for this competition.

We chose Stanford CoreNLP [11] to do most of the work. This suite of tools is freely available and ready-to-run for English, and it addresses all the basic levels of natural language processing, mainly:

- *tokenization*: identification of word and sentences boundaries, according to punctuation marks;
- *POS tagging*: identification of each word's part-of-speech (POS) category (Noun, Verb, etc.), along with a number of inflectional features (number, tense, etc.);
- *lemmatization*: identification of each word's lemma, or citation form;
- *syntactic parsing*: identification of the syntactic relationships between words in a sentence. The Stanford Parser is a dependency analyzer, and provides tagged pairwise links between syntactically related words (subject, object, determiner, etc.);
- *named entity recognition*: identification and classification of expressions that refer to a person, organization, date, quantity or location.

To avoid tweaking the main parameters of these programs, we applied a small number of modifications to the data. We replaced all `<NAME/>` XML tags (resulting from the anonymization process) with an arbitrary string, as CoreNLP could not cope with such input. We also detected (and replaced with a short arbitrary string) the message parts which contained raw (non-text) data, and would have led the most sophisticated processes (mostly the tagger and parser) to crash due to the lack of sentence delimiters. Finally, we limited the size of sentences to be processed to 50 words, discarding longer sentences which would have taken too long to parse and would probably not lead to interesting fine-grained linguistic features.

The features used for our experiment were computed on the output of these processes, with a few exceptions that are specifically indicated. The following sections give the complete list, according to the linguistic level to which they belong.

2.2 Sub-word-level features

The first level addresses linguistic characteristics that appear inside words themselves. Most of these do not need any kind of linguistic preprocessing and were in fact computed directly on the raw message text.

Character trigrams. Language models based on character n-grams have long been used for various text mining tasks, such as language identification [15]. Regarding authorship attribution, they offer many advantages: they have proven to be effective [3] and additionally, they are language independent and do not require any preprocessing (not even tokenization). Merely using character n-grams may partly capture some of the features described below: suffixes, punctuation, case, contractions, smileys, function words such as prepositions, pronouns and conjunctions, as well as some cases of alternative spelling (e.g. American vs British English, such as *-or*/*-our*, *-ise*/*-ize*).

The character trigrams occurring in the larger train set were extracted and the 10,000 most frequent ones were retained as features.

Suffixes. Features related to the morphological properties of the words can be computed both at the sub-word-level and at the word-level. In both cases, we used the CELEX database [1], a resource which provides morphological analysis for a significant fragment of the English lexicon. The morphological features computed at the sub-word-level concerned the suffixes of derived words in the CELEX parses. These features were computed directly on the lemmas. For instance, one such feature is the number of words in the message whose lemmas end in *-ion*. We only considered suffixation because suffixed words are the most frequent derivatives and suffixes are better predictors of word formation than prefixes.

More precisely, we collected from CELEX the 149 suffixes which occur at the end of at least one suffixed word. Then, for each message and each suffix, we counted the number of words which end with this suffix. Two additional features were also computed for each message: the ratio of suffixed words to total words and their overall number.

Punctuation. The rate of each regular punctuation mark, multiple marks such as `!!` and `???`, as well as space misuse in punctuation was computed. Indeed, some authors are more prone to neglect the use of punctuation marks in emails than in formal writings, or do not have an in-depth knowledge of typographic rules. Recurrent misuse of spaces may reveal a non-native speaker, e.g. a French author may repeatedly leave a blank before a colon when he writes in English. A list of 34 features corresponding to regular punctuation marks and misuse patterns was thus compiled.

Smileys. A list of possible smileys was compiled and the 6 appearing in the Test sets were selected. For each of these specific smileys, a binary feature indicated the presence or absence of this smiley in a given message. The use of smileys was expected to be more effective in capturing types of messages (formal/informal) rather than authors' style. However, the fact that some authors use some specific smileys, or no smiley at all may be a hint of authorship.

2.3 Word-level features

These features deal with formal characteristics of word units. They take advantage of the tokenization performed by CoreNLP (and therefore inherit its errors). For most features, lemmatization and POS tagging results were used, and for some of them we had to build specific lists of words or patterns, which we made available at the following URL:
http://redac.univ-tlse2.fr/projects/authorshipidentification/.

Word frequencies. According to the traditional approaches of text categorization, we computed the relative frequency of each wordform encountered in the messages. As it will be detailed in § 3, this set of features will be discarded in some parts of our experiments to focus on linguistically richer features.

Case. The proportion of all capitals tokens and tokens starting with an upper case was computed. Indeed, some authors tend to omit initial capital letters after sentence endings. Some writers have a tendency to "shout", i.e. write a whole sentence, or a part of it, in all caps. Two specific features

were dedicated to the forms i/I, as some authors use the lower case in the emails instead of the regular upper case.

Morphologically complex words. Morphological features were also computed at the word level, once again with the CELEX database. We compiled the 30,693 morphologically complex words from CELEX, leaving out the ones formed by conversion (e.g. $\text{hand}_N \rightarrow \text{hand}_V$). These words include suffixed words (e.g. *governable*), prefixed words (e.g. *anti-aircraft*) and compounds (e.g. *handbook*). Two features were computed for each message: the ratio of morphologically complex words to total words and their overall number.

Word length. Messages can also be characterized by the length of their words. For instance, the presence or absence of long words may indicate a message's technicality. Word length is also an approximate estimation of the morphological complexity of the words: longer words tend to be more complex. For each message, we counted the number of words of each length and we also computed the mean word length.

Inflections and stopwords. The inflection rate (ratio of inflected forms to lemmas) was computed and resulted in 3 features (for nouns, verbs and adjectives). The stop words rate was used as an additional feature.

Spelling errors. Spelling errors have long been identified as related to a particular author's writing specificities [12]. We compared each lowercase word in a message to a reference wordlist (from the `wamerican` Debian package), and applied a brute-force approach to check for specific spelling errors. More precisely we tested for the following errors (in order of priority): excess repeated letter, missing repeated letter, letter inversion, excess letter, missing letter, modified letter. In addition, and in cases when none of the previous tests led to a known word, we tested for word collision through automated querying of the Bing Web search engine (which, when given as input "*officetoday*" proposes to correct it in "*office today*").

The resulting features give the overall rate of misspelled words in a message, as well as specific rates for each kind of spotted error. While at first we considered letter-specific errors (e.g. repeated inversion of i and o), the lack of variety over the corpus led us to limit our features to broad categories.

Contractions and abbreviations. A list of contractions and their corresponding full forms was compiled manually and resulted in about 200 features. These forms were then divided into the following classes: positive contraction with apostrophe (*we're*), negative contraction with apostrophe (*isn't*), positive full forms (*we are*), negative ones (*is not*), standard and nonstandard "collapsed" forms (*cannot*, *gimme*), "improper collapsed" forms (*arent*, *thats*) and a class of words used specifically in spoken language (*gonna*, *kinda*, *nope*, *ya*, ...). These classes resulted in 7 additional features.

A list of 450 abbreviations was compiled manually. It contained usual abbreviations such as *ASAP* or *wrt*, some taken from SMS language (e.g. *2L8* for *too late*) and some other originating from chat rooms (e.g. *BBIAB* for *be back in a bit*). Each abbreviation resulted in an individual feature.

US/UK variants. Two lists of roughly 550 American words and 550 British ones were manually compiled, using lexicons available from the web. These lists include alternative words (*vacation/holiday*, *zip code/postcode*), spellings (*color/colour*, *vs./vs*). The vast majority of the messages were expected to be written by American authors. However, for some authors, the use of American and British words may overlap (e.g. a message in the test set contains the British *holiday* and the American *bill* and *gotten*).

Binary features denote the presence/absence of the 70 British and 170 American forms taken from the aforementioned lists that occur in the Test sets. Two features account for the total number of forms (types) from both categories and two features represent the ratio of American (resp. British) forms to the number of tokens.

WordNet. A total of 12 features were based on WordNet. Four features represent the proportion of noun (resp. verb, adjective and adverb) lemmas occurring in the messages that are known from

Princeton WordNet [6]. Four additional features (one per part-of-speech) consist of the average number of synsets the words that are known from WordNet belong to. These features roughly represent the degree of polysemy of the words used by the authors. For each message, two features represent the average depth of the noun and verb synsets in WordNet's conceptual hierarchy. Two other features denote the average minimal depth for nouns and verbs. These depth-based features are indicative of semantic specificity [9].

Named entities. As presented above, the CoreNLP suite provides a named entity (NE) recognition and classification tool [7]. Therefore, we computed the relative frequency of each NE type in a message (date, location, money, number, ordinal, organization, percent, person and time).

2.4 Sentence-level features

The next set of features addresses the syntactic level. Syntax can be approached through crude techniques, or by taking advantage of the syntactic parsing provided by CoreNLP.

N-grams of part-of-speech tags. Bigrams and trigrams of part-of-speech tags (ignoring inflection indicators) were computed for all messages. The 732 bigrams and the 1,000 most frequent trigrams (from a total of 6,598) were retained. Part-of-speech trigrams have been used in [14] as an approximation of syntactic structures, in order to detect two groups of speakers including English and Finnish emigrants to Australia, and of adult and children among the Finnish immigrants. Part-of-speech n-grams are relevant here either because they reveal differences between native and non-native speakers, and differences in the syntactic patterns used by native speakers.

Syntactic depth and complexity. The general notion of syntactic complexity can be seen as an important feature of an author's style. In addition to the simple measure of sentence length (in number of words), we measured two different parameters for each sentence in a message.

The first is simply the depth of the syntactic tree resulting from the syntactic dependencies provided by the parser (after minor transformations). Sentences with deep trees have a large number of intermediary constituents, such as complex noun phrases, subordinates, etc. We measured both the maximal depth for each message and the average depth.

The second parameter is more directly related to the output of the parser. Following [13], we measured the average and maximal distance (expressed in number of words) covered by the identified dependency links. The resulting feature is not correlated to the tree depth, but can capture another dimension of syntactic complexity [19]. As an example, a complex subject noun phrase (e.g. including a subordinate clause) results in an increased distance between the head noun and the verb.

Syntactic dependencies. From the aforementioned syntactic parsing, a list of 2,439 types of syntactic dependencies was extracted. A dependency consists of the governor and dependent's parts-of-speech (leaving out inflection indicators), linked by a given relation. The 1,000 most frequent dependencies were retained and resulted in 1,000 corresponding features denoting the ratio between the occurrences of a given dependency in a message and the total number of dependencies. We added three more general features to represent the total number of dependencies in a message, this number divided by the number of tokens and divided by the number of sentences.

2.5 Message-level features

The fourth set of features deals with the message at a global level, and addresses a variety of linguistic issues. The following features tackle the message's typesetting, discourse structures and organization, as well as general semantic phenomena such as semantic cohesion.

Message length and general typesetting. The number of tokens, sentences, lines and the proportion of blank lines were computed for each message.

Openings and closings. The most frequent opening lines were extracted and manually selected as potentially relevant. This selection resulted in 22 features including: no opening line, proper name followed or not by a colon or a comma, greetings starting with *hello/hey/hi/dear*, etc.

Potential closing patterns were extracted from the messages and manually selected. First, a list of non anonymized signatures (lower case first names and family names, initials, etc.) resulted

in 72 features. The presence/absence of some closing formulas (*thanks/thanx/(all) the best/(best) regards/take care*, etc) or of (anonymized) names resulted in 10 features. Finally, a feature was dedicated to the detection of 44 selected closing patterns, consisting of a combination of names, comma/dot, blank lines, closing formulas, etc.

Distributional semantic cohesion. This last set of features was computed in order to estimate the "semantic cohesion" of an email message. The idea is to compute semantic similarity between words in a message, so that cohesive messages (dealing with a narrow topic, and thus containing many semantically related words) can be distinguished from the ones addressing multiple subjects (with many unrelated words). This kind of measure can also be seen as an attempt to capture some of the writing behaviors of authors.

Measures of semantic similarity between words can be obtained from manually-constructed resources such as WordNet or from the data produced by distributional semantic analysis. The distributional analysis assumption is that similar words can be identified according to their occurrence in similar contexts. Thus, through the syntactic processing of a very large quantity of texts, specific techniques can automatically quantify the semantic similarity between words.

We used Distributional Memory (DM) [2] which organizes the distributional information in a third-order tensor. It has been computed on a very large corpus of diversified texts. Specifically, we used the TypeDM model[1] and its *word-by-link-word* matrix, which results from the tensor's matricization.

In particular, for each adjective, noun and verb in the matrix, we identified the 150 nearest semantic neighbors by calculating the cosine similarity. The features defining the semantic cohesion of the message were then obtained by counting the number of neighbors for each pair of lemmas found in the message. Different neighborhood sizes were considered: from 10 to 150 neighbors (in steps of 10).

3 Machine learning techniques and tools

The next step in our approach was to apply a machine-learning technique to this huge set of features. We used two different kinds of techniques, which are presented in this section.

3.1 Maximum entropy

For the author attribution tasks, we trained a single maximum entropy model [17] using the OpenNLP[2] MaxEnt library. While it is theoretically possible for a maximum entropy model to integrate thousands of heterogeneous features, both numeric and nominal, it is critical to normalize numeric features to avoid a bias in the model towards features with a larger numeric scale.

The features described above were collected as a set of CSV files. To simplify the task of normalizing, discretizing, training and evaluating based on features spread among dozens of files, we coded a software module, csvLearner[3], distributed as open source, and specifically designed for collaborative machine learning projects.

Feature normalization. For the sake of simplicity, we initially tried to unify the features into a purely nominal space by applying Fayad and Irani MDL discretization [5]. We then attempted two types of normalization: setting max value to 1.0, and setting the mean value to 0.5 (the latter to correctly normalize features with a handful of high outliers). Finally, we divided the features into several groups, each of which was normalized based on the max value of the entire group, while leftover features were normalized individually. By normalizing an entire group of interrelated features together (e.g. POS tag trigram counts), we avoided loss of information contained in their relative scale (e.g. relative count of each POS tag trigram for a given message).

Results for each of these normalization methods, when applied to SmallTrain using 10-fold cross validation, are shown in the table below. Since there is not a significant difference between max and mean normalization, we settled on max grouped normalization.

[1] http://clic.cimec.unitn.it/dm/#data
[2] http://incubator.apache.org/opennlp
[3] https://github.com/urieli/csvLearner

Method	Mean Accuracy	Std. Dev.
Raw features	19.86%	2.62%
Discretization	66.18%	2.75%
Normalization (max)	67.08%	2.96%
Normalization (mean)	67.21%	2.58%
Grouped normalization (max)	69.74%	2.05%
Grouped normalization (mean)	69.44%	2.17%

Separating the wheat from the chaff. For tasks including unknown authors, the training sets were evaluated to find a criterion for separating known and unknown authorship. One of the main advantages of the maximum entropy machine learning algorithms is that it provides a probability for each possible outcome. Various parameters were thus considered, including p_1 (best guess probability), p_2 (second guess probability), $p_1 - p_2$, z-score(p_1), z-score(p_2) and standard deviation of probabilities for all outcomes. Curiously, parameter spread for wrongly guessed messages and for messages by unknown authors was strikingly similar, and clearly differentiated from the spread for correctly guessed messages.

We finally settled on p_1 alone as providing the cleanest and simplest criterion, and selected the p_1 threshold which maximized overall accuracy on SmallValid+ ($p_1 < 0.66$ results in 'unknown') and LargeValid+ ($p_1 < 0.4$). We also submitted another run which aimed at a better precision, with respective thresholds of 0.95 and 0.75.

3.2 Rule-based learning and decision trees

For the verification tasks (Verify1, Verify2 and Verify3, each dealing with a different single author), we applied a machine learning technique that allows us to get interpretable results, and thus to identify the linguistic profile of the target author. Another objective was to consider a decision scheme that does not rely on "knowledge-poor" features (mostly trigrams and word frequencies), but on linguistically-enriched features only. This decision was not motivated by the specificity of the verification subtask, but rather by the smaller size of its data and its more specific and strict aim.

We thus switched from the heavy-duty opaque maximum entropy learner to less sophisticated learning techniques. The specific algorithm for each set was simply chosen according to the compared performances on the training data.

For the Verify1 dataset, we adopted a decision tree (C4.5 algorithm [16]) which splits the training dataset in different subsets according to the most 'informative' features, i.e. the features that maximize the information gain condition (the highest score of information gain for one feature if we use that feature to split the data). For the remaining Verify2 and Verify3 datasets, we exploited a rule-generating algorithm ('Repeated Incremental Pruning to Produce Error Reduction' or RIPPER [4]) that implements an incremental reduced-error pruning method to establish whether a rule should be pruned or not. Both classification algorithms were optimized, trained and applied in the WEKA software package[4] [8].

In both cases, the training data was constituted of the corresponding provided training sets, consisting of a few dozens messages by the target author, to which we added 1,500 messages written by a variety of different authors from the SmallTest training set.

Details. Figure 1 illustrates the decision tree adopted after training the Verify1 dataset. For better readability we do not indicate the exact thresholds for each decision node. The first node of the model corresponds to a specific typographic error (in this case, the absence of a blank after a question mark, see § 2.2). Other classificatory features are the amount of person named entities (see § 2.3), the distributional semantic neighbors extracted from the Distributional Memory (DM) (see § 2.5), the maximal depth of the syntactic tree (see § 2.4), the number of past participles as indicated by the POS parser (see § 2.3) and the apostrophe count for each message (see § 2.5).

Table 1 illustrates the rules resulting from the RIPPER model for the Verify2 and Verify3 tasks. In addition to some distributional semantic neighbors features, these datasets made use of

[4] http://www.cs.waikato.ac.nz/ml/weka

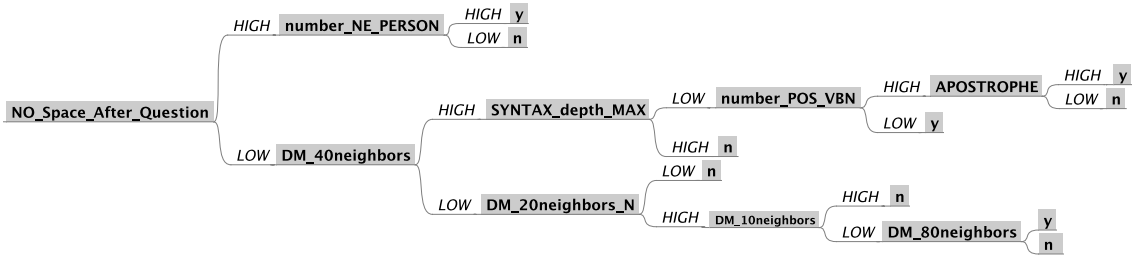

Fig. 1. Decision tree model used for the Verify1 task (y = target author; n = other author)

a mix of word-level features: the presence of words with specific morphological suffixes (§ 2.3) and the frequency of a specific POS trigram (§ 2.4).

Verify2 – RIPPER rules:

1. **if** $DM\_90neighbors\_NORM \geq 0.00493$ **and** $DM\_80neighbors \leq 9$ **and** $APOSTROPHE \leq 0$ **then** Y
2. **if** $DM\_20neighbors\_NORM \geq 0.0173$ **and** $COLON \geq 0.0090$ **and** $DM\_10neighbors \leq 28$ **then** Y
3. **otherwise** N

Verify3 – RIPPER rules:

1. **if** $DM\_30neighbors\_NORM \geq 0.00724$ **and** $Morpho\_SUFFIXES\_NB \leq 7$ **then** Y
2. **if** $number\_POS\_SYM \geq 0.175439$ **then** Y
3. **if** $DM\_20neighbors\_NORM \geq 0.03649$ **and** $DM\_130neighbors\_NORM \leq 0.197287$) **then** Y
4. **if** $DM\_20neighbors\_NORM \geq 0.001346$ **and** $DM\_30neighbors \leq 6$ **and** $Trigrams\_POS = DT - NN - . \geq 0.009346$ **then** Y
5. **otherwise** N

Table 1. Rules learned for the Verify2 and Verify3 tasks

This tree and these sets of rules gave us positive feedback concerning the usefulness of a variety of linguistic features. They can also tell us specific information about each target author. For example, author 1 tends to forget blanks after question marks, and uses a high number of people's names. However, the different semantic distribution scores seem to have the highest discriminative power, and yet are not very easy to interpret in terms of an author's general writing style.

Table 2 summarizes the scores obtained with the aforementioned models with respect to the three verification tasks. The scores concern the positive (target author) class, obtained on average over a 10-fold cross-validation. We initially were quite satisfied with these results, that seemed on a par with the ones obtained with the maximum entropy method, although on a very different task.

4 Results and discussion

The overall results of our different runs are the following: we apparently did very well for the attribution tasks, but failed for all three verification tasks. Detailed values are reported in Table 3. For the Large+ and Small+ tasks, it was indeed the low-threshold version that gave the best

Task	Precision	Recall	F-score
Verify1	0.824	0.667	0.737
Verify2	0.636	0.509	0.566
Verify3	0.609	0.596	0.602

Table 2. Training scores for each verification task (10–fold cross-validation)

results in terms of F-score: with higher probability thresholds, the precision gain was considerable but did not counterbalance the loss in recall.

Task	Rank	Precision	Recall	F-score
Large	3/18	0.620	0.444	0.459
Large+	1/13	0.688	0.267	0.321
Small	1/17	0.662	0.451	0.475
Small+	1/13	0.737	0.161	0.193
Verify1	8/10	0.091	0.333	0.143
Verify2	8/10	0.100	0.200	0.133
Verify3	5/10	0.083	0.250	0.125

Table 3. Resulting ranks and macro-average scores per task

Many things can be learnt from these results.

The first thing is that the good results we obtained for the attribution tasks may well be related to the linguistically-rich features we used in addition to more traditional word/trigram frequencies. During the training phase, we indeed observed that these features increased the estimated efficiency of the maximum entropy learner. However, we still have to investigate the amount of information added by individual features or feature sets. As noted above, this estimation is much more difficult for this kind of method than for symbolic learning, as the information about the model is only accessible through a set of weights and require a detailed statistical analysis. However, we intend to test a larger number of features combinations in order to assess their exact contribution to the final results.

Another good point for our method is the apparent reliability of the probability score provided by the maximum entropy algorithm: this appears to be a good predictor that allowed our method to rank very high with the unknown author tasks. This estimation is, to our eyes, a clear advantage of this learning technique, that can have several positive implications in other tasks where maximum entropy has proven to be efficient (such as parsing).

The last issue raised is our failure in the verification tasks. It may be linked to some of the following causes: abandonment of the poorer features, switch from maximum entropy to rule-based techniques, and the fact that author verification is indeed a very different task from attribution. Each of these hypotheses will now require a specific investigation that should lead us to a better understanding of both the tasks, the applicability of the techniques and the relative advantage of our higher-level features.

As we have already noted, our main concern as computational linguists is to evaluate the effectiveness of more sophisticated language processing methods and features. In too many areas, linguistically-rich approaches have unfortunately not proven (yet) their efficiency when compared to heavy statistical methods. We hope that our work will contribute to a better understanding of how, when, and which linguistic knowledge has to be used.

Acknowledgements

As noted in the introduction, our success in this competitive task is the result of a team that extends far beyond the authors of this paper. Therefore we sincerely wish to thank for their

insights, proposition of features and encouragements the following people (in alphabetical order): Clémentine Adam, Cécile Fabre, Bruno Gaume, Mai Ho-Dac, Anna Kupść, Marion Laignelet, Fanny Lalleman, François Morlane-Hondère, Marie-Paule Péry-Woodley and Nikola Tulechki.

References

1. Baayen, R.H., Piepenbrock, R., Gulikers, L.: The CELEX lexical database (release 2). CD-ROM (1995), linguistic Data Consortium, Philadelphia, Penn.
2. Baroni, M., Lenci, A.: Distributional memory: A general framework for corpus-based semantics. Computational Linguistics 36(4), 673–721 (2010)
3. Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. In: Proceedings of the Third Annual Symposium on Document Analysis and Information Retrival. pp. 161–175 (1994)
4. Cohen, W.W.: Fast effective rule induction. In: Twelfth International Conference on Machine Learning. pp. 115–123 (1995)
5. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: IJCAI. pp. 1022–1029 (1993)
6. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press (1998)
7. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005). pp. 363–370 (2005)
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. ACM SIGKDD Explorations Newsletter 11(1) (2009)
9. Heylen, K., Peirsman, Y., Geeraerts, D., Speelman, D.: Modelling Word Similarity: an Evaluation of Automatic Synonymy Extraction Algorithms. In: Proceedings of the Sixth International Language Resources and Evaluation (LREC'08). Marrakech, Morocco (2008)
10. Juola, P.: Authorship attribution. Foundations and Trends in Information Retrieval 1(3), 233–334 (2006)
11. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Meeting of the Association for Computational Linguistics. pp. 423–430 (2003)
12. Koppel, M., Schler, J.: Exploiting stylistic idiosyncrasies for authorship attribution. In: IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis. pp. 69–72 (2003)
13. Mothe, J., Tanguy, L.: Linguistic features to predict query difficulty. In: Proceedings of the ACM-SIGIR workshop on Predicting query difficulty - methods and applications. pp. 7–10. Salvador de Bahia Brazil (2005)
14. Nerbonne, J., Wiersma, W.: A measure of aggregate syntactic distance. In: Proceedings of the Workshop on Linguistic Distances. pp. 82–90. LD '06, Association for Computational Linguistics, Stroudsburg, PA, USA (2006)
15. Peng, F., Schuurmans, D., Keselj, V., Wang, S.: Language independent authorship attribution using character level language models. In: Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics, EACL. pp. 267–274 (2003)
16. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA (1993)
17. Ratnaparkhi, A.: Maximum entropy models for natural language ambiguity resolution. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA (1998)
18. Stamatatos, E.: A survey of modern authorship attribution methods. Journal of the American Society for Information Science and Technology 60(3), 538–556 (2009)
19. Tanguy, L., Tulechki, N.: Sentence Complexity in French: a Corpus-Based Approach. In: Proceedings of IIS (Recent Advances in Intelligent Information Systems). pp. 131–145. Krakow, Poland (2009)
20. Zheng, R., Li, J., Chen, H., Huang, Z.: A framework for authorship identification of online messages: Writing-style features and classification techniques. Journal of the American Society for Information Science and Technology 57(3), 378–393 (2006)